Certification Requirements for High Assurance Systems

Gordon M. Uchenick
Senior Mentor/Principal Engineer
Objective Interface Systems, Inc.

and

W. Mark Vanfleet
Senior Cryptologic Mathematician/ Senior INFOSEC Analyst
National Security Agency

The Joint Services Systems & Software Technology Conference

What is High Assurance?

- To the FAA:
  - One failure per 10^9 (1 Billion) hours of operation
    - How long *is* a Billion hours? Do the math!
      - $1{,}000{,}000{,}000\ \text{hours} \times \dfrac{1\ \text{day}}{24\ \text{hours}} \times \dfrac{1\ \text{year}}{365.25\ \text{days}}$
      - 114,077 *YEARS!*

- For National Security Systems processing our most valuable data under most severe threat:
  - Failure is *Unthinkable*

- ***How do we implement systems that we can trust to be this robust?***

Why Software Certification?

- In environments requiring High Assurance, the consequences of failure can be loss of human life
- To ***PREVENT*** failure, software should be rigorously inspected by domain experts independent of developers
  - "Fail-first/Patch-later" not acceptable – the damage is done
    - Service Packs/Critical Updates, for example
- Testing alone is not enough
  - Rigorous inspection requires a rigorous methodology
  - Based on "claims substantiated by evidence"
  - Testing without evidence is ineffective
    - Processor faults, for example
  - Need to verify more than just the code
    - The development process itself
    - Lifecycle control
    - Flaw remediation infrastructure

Certification Standards are Domain Specific

- Safety Critical Systems
  - RTCA DO-178B, *SOFTWARE CONSIDERATIONS IN AIRBORNE SYSTEMS AND EQUIPMENT CERTIFICATION*
- Security Critical Products
  - ISO-15408, *COMMON CRITERIA for INFORMATION TECHNOLOGY SECURITY EVALUATION*
- Security Critical Systems
  - DIACAP: *DoD INFORMATION ASSURANCE CERTIFICATION AND ACCREDITATION* ***PROCESS***
    - Replaced DITSCAP: *DoD INFORMATION TECHNOLOGY SECURITY CERTIFICATION AND ACCREDITATION* ***PROCESS***
  - DCID 6/3-JAFAN 6/3: *PROTECTING SENSITIVE COMPARTMENTED INFORMATION WITHIN INFORMATION SYSTEMS*

Levels of Certification

- DO-178B: Level E (low) through Level A (high)
  - Level E: Failure has no effect: Entertainment systems
  - Level A: Failure is catastrophic: Fly-by-wire controls
- Common Criteria: Evaluation Assurance Level 1 (low) through 7 (high)
  - EAL1: Functionally Tested
  - EAL7: Formally Verified Design and Tested
- DCID 6/3: Protection Level 1 (low) through 5 (high)
  - PL1: All users have clearance, approval, and need to know
  - PL5: Some users are uncleared

Which Level, and Why?

- DO-178B: Pilots and engineers analyze effect of software component failure upon total aircraft safety
  - Fault tree analysis: “If this one fails, it’s a C, if that one fails, it’s a C, if they both fail, it’s an A”
- DCID 6/3: Based upon clearance, approval, and need to know
- Common Criteria: Based upon value of information and threat potential of attackers
  - Definition of terms in Chapter 4, *Information Assurance Technical Framework*

Information Value	Threat Levels						
	T1	T2	T3	T4	T5	T6	T7
V1	SML1 EAL1	SML1 EAL1	SML1 EAL1	SML1 EAL2	SML1 EAL2	SML1 EAL2	SML1 EAL2
V2	SML1 EAL1	SML1 EAL1	SML1 EAL1	SML2 EAL2	SML2 EAL2	SML2 EAL3	SML2 EAL3
V3	SML1 EAL1	SML1 EAL2	SML1 EAL2	SML2 EAL3	SML2 EAL3	SML2 EAL4	SML2 EAL4
V4	SML2 EAL1	SML2 EAL2	SML2 EAL3	SML3 EAL4	SML3 EAL5	SML3 EAL5	SML3 EAL6
V5	SML2 EAL2	SML2 EAL3	SML3 EAL4	SML3 EAL5	SML3 EAL6	SML3 EAL6	SML3 EAL7

Certification and Level of Effort

- DO-178B: Objectives by Level

Level	Objectives	Independent Obj
A	66	25
B	65	14
C	57	2
D	28	2

- Common Criteria Levels:
  - EAL1: Functionally Tested
  - EAL2: Structurally Tested
  - EAL3: Methodically Tested and Checked
  - EAL4: Methodically Designed, Tested, and Reviewed
  - EAL5: Semiformally Designed and Tested
  - EAL6: Semiformally Verified Design and Tested
  - EAL7: Formally Verified Design and Tested

OBJECTIVE INTERFACE

Upper Common Criteria EALs (Operating System Examples)

- EAL4
  - Lots of examples – Windows, Linux, Solaris, etc.
  - "EAL4 permits a developer to gain maximum assurance from positive security engineering based on good commercial development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources. *EAL4 is the highest level at which it is likely to be economically feasible to retrofit to an existing product line."*
- EAL5
  - One example – BAE XTS 400
  - "EAL5 permits a developer to gain maximum assurance from security engineering based upon rigorous commercial development practices supported by moderate application of specialist security engineering techniques. Such a TOE will probably be designed and developed with the intent of achieving EAL5 assurance."
- EAL6
  - One example nearing completion – Green Hills Integrity-178
  - "EAL6 permits developers to gain high assurance from application of security engineering techniques to a rigorous development environment in order to produce a premium TOE for protecting high value assets against significant risks."
- EAL7
  - Nothing even on the radar
  - "EAL7 is applicable to the development of security TOEs for application in extremely high risk situations and/or where the high value of the assets justifies the higher costs. Practical application of EAL7 is currently limited to TOEs with tightly focused security functionality that is amenable to extensive formal analysis."

- Robustness = Strength of Function **+** Assurance
  - Strength of Function
    - How high is the fence?
    - How strong is the fence?
  - Assurance
    - How confident is the buyer that the fence meets all of the builder’s claims?
  - Stakeholder must determine the appropriate level of robustness
- Basic Robustness
  - Assurance approximates EAL3+
  - *Consistency Instruction Manual For Development of US Government Protection Profiles for Use in Basic Robustness Environments*
- Medium Robustness
  - Assurance approximates EAL4+
  - *Consistency Instruction Manual For Development of US Government Protection Profiles for Use in Medium Robustness Environments*
- High Robustness
  - Assurance approximates EAL6+
  - Only guidance is *U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness,* still in evaluation

Medium Robustness

- At Medium Robustness, primary mechanism for achieving assurance is testing
  - Attack-Oriented, Brute Force and Random, Fault and Vulnerability-Oriented, Security Fault Injection, Usability, Performance, Compliance, Reliability, Operational, Functional
- Testing can't be exhaustive even for trivial systems
- Testing can only be as comprehensive as the artifacts upon which it is based
- Testing coverage is limited
  - Testing profile may not be relevant to the way that the system is actually used
  - Can't find malicious functionality – back doors, Trojan horses, etc.
  - Penetration testing by brute force, hunting for any vulnerability
  - Can't prove *absence* of faults or vulnerabilities

High Robustness

- At High Robustness, primary mechanism for achieving assurance is mathematical proof of correctness
- Mathematical statement of security policy completely identifies unauthorized events
- Mathematical modeling finds all functional errors
  - Independent of test script completeness
- Theorem proving exposes assumptions and hidden assumptions
  - Assumptions are the basis for safe and secure embedment (more about this later)
- Testing is based on
  - Functional (same as Medium Robustness)
  - Foundational, based on formal model
  - Penetration testing, attempts to invalidate assumptions, is provably complete

Safe Systems vs Secure Systems

- High assurance safe systems do what they are supposed to do
  - A benign user is assumed
  - The pilot may make mistakes which the software has to withstand, but he really wants to fly the airplane safely
- High assurance secure systems do what they are supposed to do **AND NOTHING ELSE** ***AND WE CAN PROVE IT***
  - A malicious user is assumed
  - From the IATF: "Extremely sophisticated adversary with abundant resources *(nationally funded intelligence agency with hordes of geeks)* who is willing to take extreme risk (e.g., nation-states in time of crisis)
    - *Italic text is our commentary*

- Safety and Security Certifications have much common ground
  - Does the software meet its requirements?
  - Is it well designed?
  - Is it well implemented?
  - Is it well tested?
  - Are appropriate life cycle controls in place?
  - Are plans in place to address flaws as they are found?
  - Are plans in place to maintain assurance as the product or its environment evolve?
- Reference: Alves-Foss, Rinker, and Taylor; *Towards Common Criteria Certification for DO-178B Compliant Airborne Software Systems,* January, 2002, University of Idaho, http://www.cs.uidaho.edu/~jimaf/papers/compare02b.htm

Reuse of High Assurance Certifications

- Certifications are expensive and we must find a way to reuse them
- Medium robustness certifications are problematic to reuse if there is any configuration or environmental change
  - “Delta evaluation” is a judgment call
  - Can only be certain if the entire evaluation is redone
- High Robustness evaluations are more expensive “up front” but significantly reduce total ownership cost because certifications can be safely and securely reused
- Embedment: High robustness evaluation produces a complete list of assumptions, both explicit and hidden, about the environment
- *Tests that validate environmental assumptions enable safe or secure use of previously certified components*
- *Effects of system update are completely modeled, eliminating “fixed this, broke that.”*

High Robustness and Composability

- Composability: Things retain their original properties when they are combined with other things
- *Composability is the key to realizing the benefits of COTS*
- Medium robustness certification of a component gives us no useful information about the sensitivity to (infiltration) or the generation of (exfiltration) side effects
- High robustness certification results in a complete list of assumptions that can be verified when components are combined into a system
- *Predictable behavior when systems are integrated is the key enabler to interoperability and open system architectures that are actually open*
- Composability enables Compositionality: The whole is greater than the sum of its parts
  - Example: Why everyone is interested in MILS